

Koala K.I.L.L.E.R. Documentation

By Milasoft64.Itch.IO

- 1) Introduction
- 2) Program One (simple image display)
- 3) Program Two (Koala intros)

Introduction

What is Koala KILLER?

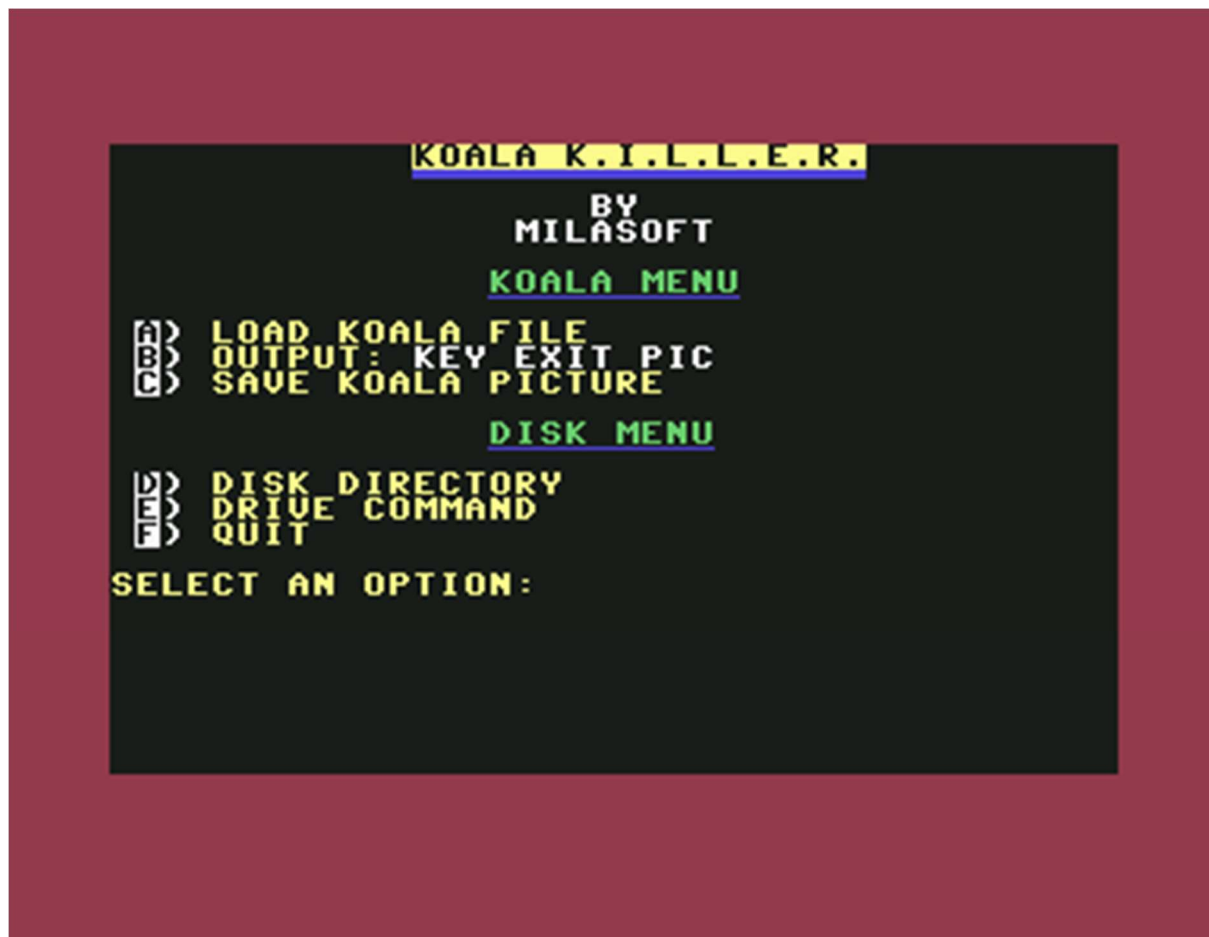
Koala K.I.L.L.E.R. stands for Koala Image Linker and Loader, so technically “Koala Koala Image Linker and Loader” rather than just “K.I.L.L.E.R.” which would scare away potential users. 😊

What does this utility do?

There are two different programs available for you, both offering their own options ranging from displaying a Koala image, to creating a game loader, to designing your own intro.

PROGRAM ONE

Program one is the basic Koala cruncher, an improved version based on Brian Conrad’s original code. It’s designed for displaying Koala images and nothing else.



A) **Loads a Koala image.** These are the standard 40 block Koala files.

You can use the "?" and "*" as wildcards. For example "?picture.kla" will load "Apicture.kla". If the image is named "my photo.kla" you can use in "my p*" as another example.

The program will not allow you to load a file that isn't in the proper Koala format.

B) **OUTPUT :** This defines the type of file output you'd like.

Key Exit Pic – This is much like the original Koala cruncher. It loads and displays a Koala image, and exits upon keypress.

Endless Pic – This is the same as the above option but it doesn't allow you to exit the image display. It's designed more for scene releases that you'd prefer to have a professional appearance.

200 Block Loader – This option will create a Koala image loader for you. After you’ve loaded a Koala image, and selected this feature, you’ll be asked for the filename of the program you wish to load. It works with files up to 200 blocks in size. The Koala image will remain on the screen as the file loads in, much like the Ocean tape loaders used to do. 😊

C) SAVE KOALA PICTURE : This writes the compressed file to disk in a RUNable format. You may want to further compress the file with Exomizer.

Technical information:

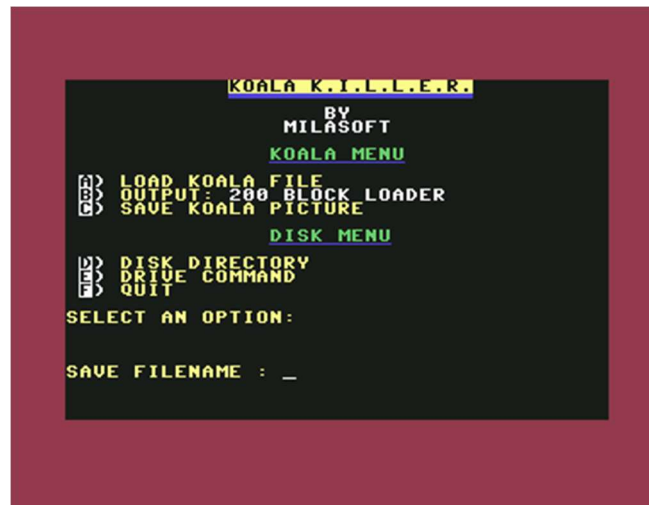
The bitmap data for the 200 Block Loader format is stored in memory at \$E000-\$FE00. The character data is saved at \$DC00. For this reason, there *might* be some compatibility issues with machines using cartridges. The loader places itself out of the way at \$033C in the cassette buffer.

The bitmap data for the “display only” Images (without a loader) is placed in memory at \$5C00 to about \$6000.

If the menu system looks familiar, it should. It was designed in homage to the Fast Hack 'Em utility.

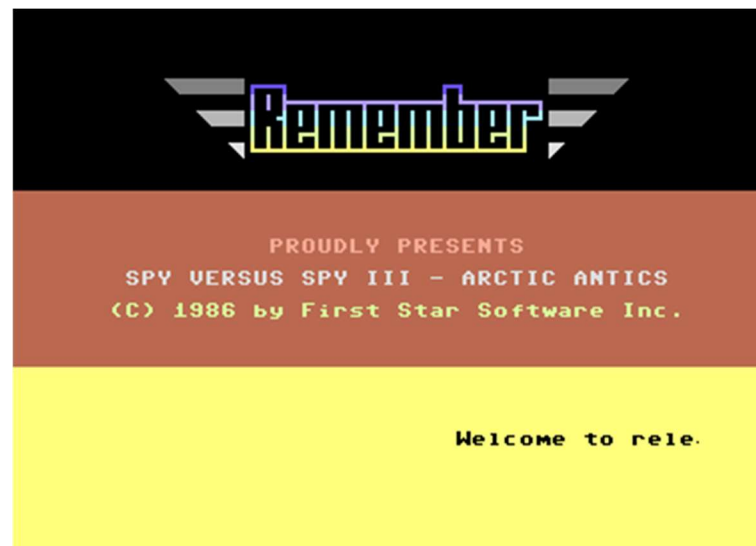


(Image information shown after loading a Koala, the option to create an endless photo)



(Ready to save the compressed file)

The loader is best demonstrated using actual hardware or the Vice Emulator. If you use the CCS Emulator, the loading is instantaneous and you'll never see the image.

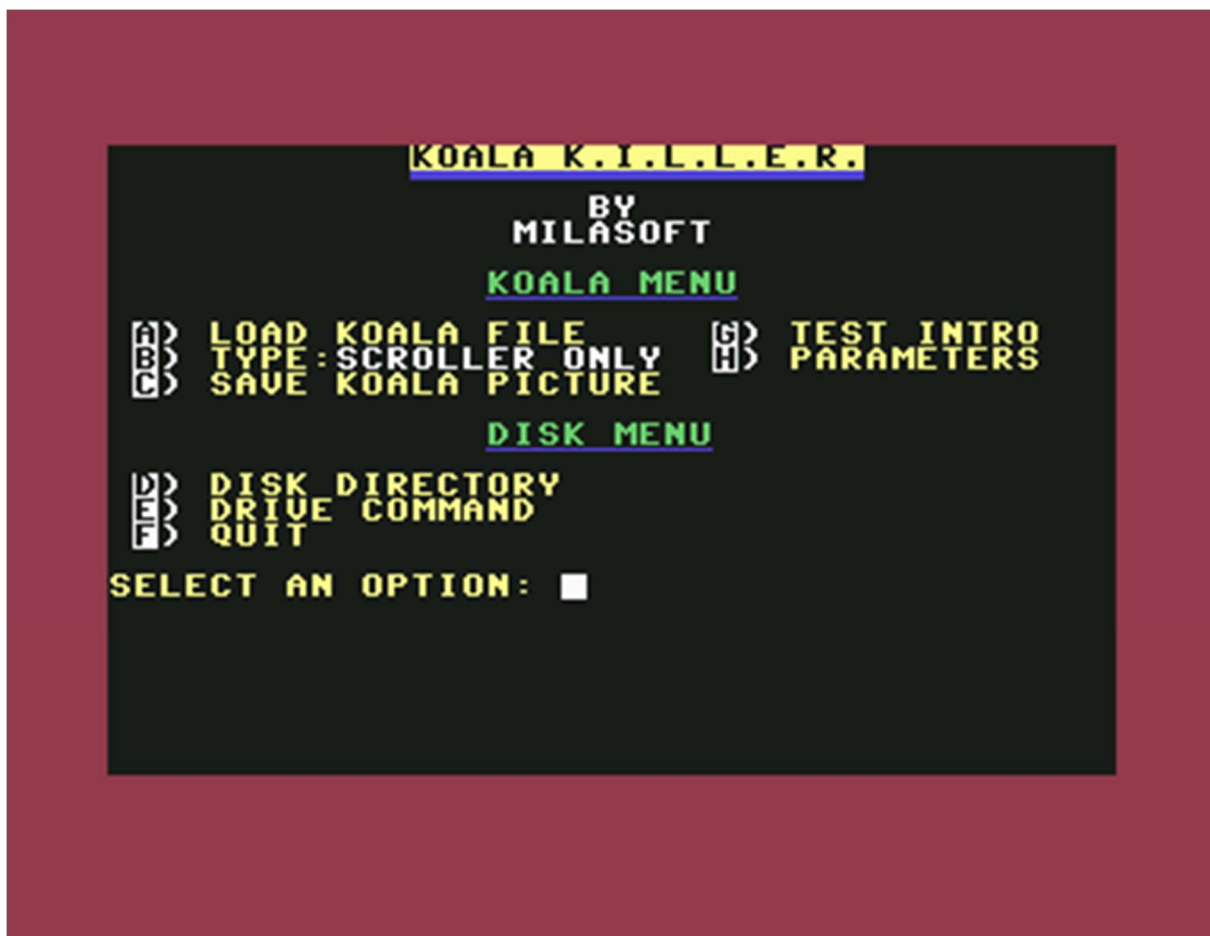


(sample Koala image shown while loading Spy Vs Spy)

PROGRAM TWO

Program two is a little more robust – and complicated. It allows you to expand upon the compressed photos you created with program one. With Program Two, you can design intros for your own projects or for existing programs. This includes having a sprite scroller (even in the upper or bottom border) as well as playing music.

The music must be in the range from \$4000-\$5BFF. Most music for the 64 is found in the range of \$1000 to \$1FFF. This memory is reversed for the compressed Koala data and not available.



The difference here is the addition of the TEST INTRO option and PARAMETERS.

The process remains the same:

A) Load a compressed Koala file

B) Select the output type (SCROLLER ONLY or FILE LOADER). Scroller only will display a Koala image along with your scrolling message (optional music). The File Loader will do the same but will load a program upon pressing Space. It acts like a game intro.

The difference between Program One and Two: Program One shows the image WHILE THE GAME LOADS. Program Two shows the image only before the game loads.

G) Allows you to view the result before saving it.



Behold, the parameter menu:

F1) Toggles expanded Y sprites. When off, regular sized sprites are used for the scroll. When on, double height sprites are used.

F3) Adjusts the color flash rate for the scroll.

F5-F6) Selects the Y position where you want the scroll to appear. The sprite on the screen provides a visual indicator of where the scroll will be. You can actually have the scroll appear in both upper and lower borders.

E) Edit the scroll text (240 characters should be enough?), ending it with an @ sign. Press Return to exit the editor.

C) Selects the color flash pattern for the scroll (solid black isn't visible but it follows the solid white).

M) Opens the Music menu.

B) Opens/closes the borders. I will explain how this works. If you want the scroll to appear on the normal screen, leave this alone. If you'd prefer to have the top/bottom borders open (and place the scroll in this area) then press B. The only use for this is if you want to place the scroll in this area. The status of the border when you press X to exit determines if the borders are open or closed in the final file saved to disk.

NOTE

Program Two will ONLY work with compressed Koala images created with Program One, or with the original Koala Cruncher by Brian Conrad. It won't work with 40 block Koala files nor will it work with the 200 block file loader made with Program one. The reason for this is that the images for the 200 block loader differ from the standalone images. Program two only works with Koala data at \$6000.

Technical Details:

Compressed Image	\$0801-\$3000
Character Set	\$3000-\$3200
Intro Code	\$3200-
Music	\$4000-\$5BFF
Sprite Pointers	\$5FF8-\$5FFF
Bitmap Data	\$6000-\$8000

Reminder – Program Two will only work with a compressed Koala picture (a file that you load, RUN, and it displays the Koala image) made using Program One or the original Koala cruncher program by Brian Conrad. It won't work with raw .KLA files.

Q: Can I change the character set used in the scroll?

A: Yes, and no. There is no option to import a character set at this time however if you have one that you'd like to use, you may break into a machine language monitor and load one into the memory area of \$3000-\$31FF.

Music Menu



The music menu allows you to load music in a binary format and define the Init and Play addresses. Most every music file from a demo, or music player, comes in a format compatible with this utility.

Music files will generally have an address to initialize the music (reset it) and another to play it.

Some music will have multiple songs or effects which can be selected by loading the accumulator with a value first. Example:

```
LDA #$00 ; first song  
JSR $4000
```



```
LDA #$01 ; second song
JSR $4000
```

And of course, JSR \$4003 to play the music. This program already initializes the A register to zero for you. The only requirement is that the music begins in the area of \$4000 to \$5BFF.

Example: If the music starts at \$5000 and ends at \$5A00, it will work! You use the I and P keys to select the Init and Play addresses in hex. Once a music file has loaded, and the Init/Play defined, you can test the song from within the utility.

Q: Can music be relocated?

A: Yes! If you have a music composer program or relocater utility, you certainly can. Another (more difficult method) is to manually change all of the addresses and zero page pointers but this is tedious. If you have any questions on music relocating, I'd highly recommend you reach out to Richard of The New Dimension.

<http://www.tnd64.unikat.sk/>



(the music menu when a music file has been loaded)

MOFT64.ITCH.IO MIL



MOFT64.ITCH.IO MIL

If you have any questions or comments, reach out to me on Lemon64.com

Greetings to Richard Bayliss, Alf Yngve, David Bradley, Schema, TPUG, Robin from 8-Bit Show and Tell, Laxity, Solo1870, Codetsu, Endurion, .mad., DDT, Slaygon, Ziona.